Bolt Beranek and Newman Inc.

(12)

LEVEL II

Report No. 4752

(14) BBN-4752-Vol-1

AD A106102

(6) **Application of Symbolic Processing to Command and Control: An Advanced Information Presentation System.**

Annual Technical Report No. 2
Volume I: Overview.

(10) F. Zdybel, J. Gibbons, M. Greenfeld, and M. Yonke
Frank Jeff Norton Martin

(9) Annual technical rept. no. 2 1980-1981,

(11) August 1981

(12) 41

(15) N0039-79-C-0316,
ARPA Order-3740

Prepared for:
Defense Advanced Research Projects Agency

JCD

81 10 26 086

060100

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A106102 / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) APPLICATION OF SYMBOLIC PROCESSING TO COMMAND & CONTROL: AN ADVANCED INFORMATION PRESENTATION SYSTEM. Vol.I: Overview, ~~Vol.II: Knowledge Base, and Vol.III: Program Source Files~~	5. TYPE OF REPORT & PERIOD COVERED Annual Technical Report 1980/1981
	6. PERFORMING ORG. REPORT NUMBER BBN Report No. 4752
7. AUTHOR(s) Frank Zdybel, Jeff Gibbons, Norton Greenfeld, and Martin Yonke	8. CONTRACT OR GRANT NUMBER(s) N00039-79-C-0316 ARPA Order No. 3740
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 10 Moulton Street Cambridge, MA 02238	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency Information Processing Techniques Office Arlington, VA 22209	12. REPORT DATE August 1981 / 13. NUMBER OF PAGES ~~450~~
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Electronics Systems Command Program Code 9D30 Washington, DC 20360	15. SECURITY CLASS. (of this report) Unclassified / 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE ----------

16. DISTRIBUTION STATEMENT (of this Report)
Distribution of this document is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
Distribution of this document is unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Information Presentation, Graphics, Knowledge Representation, Structured Inheritance Network, KL-ONE, Artificial Intelligence, Symbolic Processing, Computational Epistemology, Command and Control, AIPS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This report describes the work performed in the second year of the three-year contract to explore the application of symbolic processing to command and control (C2); specifically, the graphics interface between the C2 user and a complex C2 decision support system.

In Volume I, the goals and approaches used in the design of the prototype system, AIPS (Advanced Information Presentation System), are discussed, as → next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060100

JCB

well as the year's efforts to extend the prototype. An overview of the current AIPS system is also provided.

Volume II contains the complete AIPS knowledge base. This document provides the fully-inherited structure that the system sees during operation.

Volume III contains the programs that manipulate the knowledge base and provide the active behavioral component of the system.

Report No. 4752

APPLICATION OF SYMBOLIC PROCESSING TO COMMAND AND CONTROL:

AN ADVANCED INFORMATION PRESENTATION SYSTEM

Frank Zdybel, Jeff Gibbons, Norton Greenfeld and Martin Yonke

September 1981

Annual Technical Report #2
Volume I - Overview

Prepared by:

Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, Massachusetts 02238

Prepared for:

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

This research has been supported by the Defense Advanced Research Projects Agency under Contract N00039-79-C-0316, ARPA Order No. 3740, and monitored by the Naval Electronics System Command (Program Code 9D30).

TABLE OF CONTENTS

LIST OF FIGURES

1. RETROSPECTIVE

This report describes the second year of a three-year contract to explore the application of symbolic processing to Command and Control (C2). Specifically, we are concerned with the graphics interface between the C2 user and a complex, C2 decision support system. A description of the goals and approaches of our previous research can be found in [Zdybel et al., 1980a].

Our attention during this last year focused on a redesign of our prototype system (AIPS). This redesign pursued a more general concept of information presentation, as well as an improved architecture. Previously, we viewed a presentation as a mapping from domain objects onto display objects. In our current work, we view a presentation as a description of how arbitrary domain information is treated by or becomes involved in the production of display objects.

We therefore constructed a new KL-ONE hierarchy of abstract presentation types. We also developed a new descriptive hierarchy of display objects, including complex composite display entities, and a method for characterizing arbitrary chunks of domain information.

The internal behavior of the system has also been redesigned, and there are now distinct epochs in which the basic structure of the presentation is derived; in which two-

dimensional layout of the specified display objects is performed; and in which the finalized description is used to drive the generation of the actual display. Each of these major activities involve specialized inferencing and entrain their own particular research problems. In addition to prototype software which embodies a first attempt on some of these problems, we have implemented major tools for programming in KL-ONE, including a "derive role values as needed" system, the CKLONE database building system, and message passing control structures that enable an "object-oriented" approach to describing system behavior.

We used these tools within our revised conceptual framework to build a "starter set" of graphic presentation formats. This currently includes tabular displays and maps, including some specializations such as NTDS-MAP. In the current prototype we have described screen windows in an elaborate fashion, but are still dissatisfied. We constructed a small domain world description in KL-ONE and used it to separately debug the various pieces of the system (i.e. the format derivation, the layout process, and the graphic realization).

During this period, we also designed (but did not implement) a window-oriented user executive. This important sub-system provides for the proper interpretation of user input relative to the various windows in view, and relative to the concurrently executing processes that are capable of accepting input. We did

not attempt to build even a stripped down version, as any Interlisp-10 implementation would necessarily have been DECSystem20-dependent, and would therefore have to be largely redone for Interlisp-Jericho.

In more than one dimension, we have hit the limits of Interlisp-10 and cannot continue programming in that environment. The current AIPS has been carefully designed to minimize memory space used while still keeping its knowledge base declarative as much as possible. But even with considerable attention paid to this problem, the prototype can only be tested in pieces -- it will not fit on the Decsystem20 in its entirety.

The delivery of the Jericho symbolic processors (which at this writing has already commenced) will relieve these constraints. Our attention for the immediate future will be concentrated on the transition of the current AIPS prototype to Interlisp-Jericho.

During this reporting period, we presented a paper describing AIPS at the Thirteenth Hawaii International Conference on System Sciences, reprinted in Data Base [Yonke et al., 1980a]; we presented a discussion of requirements for user-support environments for knowledge representation languages at the Association for the Advancement of Artificial Intelligence (AAAI) conference in August [Zdybel, 1980]; and we have written two other papers clarifying the AIPS concept. Both have been

accepted: one will be presented at the Seventh International Joint Conference on Artificial Intelligence (Vancouver, 24-28 August) [Zdybel et al., 1981a] and the other will be presented at the Computer Graphics 81 International Conference (London, 27-29 October) [Zdybel et al., 1981b].

Lately, we have seen indications in the literature of increasing interest in the area of information presentation, as we see it. The Computer Corporation of America now has an active project to provide an information presentation front-end to its Spatial Data Management System. Xerox PARC indicates that its CEDAR programming environment will include an information presentation front-end for its DBMS component. There is a doctoral thesis on the topic in progress at the MIT Laboratory for Computer Science. There is even a session at the upcoming SIGGRAPH '81 conference entitled "Information Presentation", although the traditional graphics research community attaches a more restricted meaning to the phrase. These are all indications that the concept is gaining currency in the research and development community. We expect to see a variety of attempts at "information presentation" systems in the next few years.

In order to provide a comprehensive overview of the current AIPS system we have included the Computer Graphics 81 paper in the next chapter. Volume two of this report contains the complete AIPS knowledge base. This provides the fully-inherited structure that the system sees during operation. It is this

knowledge base, plus the procedural attachments contained in the program files, that constitute AIPS and determine its behavior.

Finally, in volume three of this report are the files that constitute the system. While this is not usually perspicuous, they have been included for those readers who wish to study the system in detail. Since AIPS is a knowledge-based system, its structure is embodied in the knowledge base: the code which initializes and uses that knowledge base provides the complete description of the system.

2. A CHARACTERIZATION OF AIPS

Graphic display of information plays a major role in many man-machine systems because it exploits the high-bandwidth visual channel. It is particularly useful where the task of the human component is to perceive patterns in large collections of data (e.g. a cardiogram) or in limited time (e.g. a flight attitude display). In some applications, such as military Command and Control (C2), display requirements are subject to change without notice and it is important that the user have interactive control over display content and format.

Unfortunately, a good interactive graphic interface is neither cheap nor simple. Currently such interfaces must be custom-built (generally at a major portion of the total svstem cost) and can offer only limited flexibility. This is most bothersome in composite systems which unite several interactive programs: each sub-system that uses graphics must duplicate the display generation function in its own domain-dependent terms, and it is difficult to arrange for displays that combine information from more than one source. The problem seems to be that, while we have tools (i.e. "graphics languages") for controlling the surface structure of display behavior, little has been done to abstract the display generation function in its entirety into a multi-use tool: an information presentation system.

An information presentation system automatically generates graphic displays according to partial specifications which are phrased mainly in terms of what information is to be depicted rather than in terms of how the display is to be drawn (e.g. "Display the locations of these objects"). It must, however, be prepared to bow to format specifications (e.g. "Display the names and locations of these objects as a table"), and in some cases to specifications about display appearance as well (e.g. "Display the locations of these objects as triangles on a map"). In summary, an information presentation system should free its clients from having to specify desired displays in terms of graphic entities and relationships without forcing them to abdicate control of the display generation function. This formidable prescription can never be completely filled, but we feel it can be met to a worthwhile extent.

Abstraction of the display generation function into an information presentation system confers many side-benefits because the display function can now be enhanced and elaborated in ways that would be impractical or uneconomical in a single use interface. For example, a complete treatment of display windows and window-relative graphic input becomes worthwhile [Teitelman, 1977]. The interface by which the end-user controls the display function can be built up, perhaps even to the point of allowing natural language specification of displays. A library of display formats can be accumulated. A high degree of sensitivity to the

human end-user can be built in. Not only can display formats be tailored to suit the individual, but many other details of graphic presentation such as font selection, highlighting methods and the use of color can be accommodated to individual needs. Not least important, an information presentation system is a place to embody a consistent set of decisions about the human factors of graphic display.

In a composite system, information presentation centralizes the display generation function, leading to two important benefits. First, the end user experiences a more uniform treatment of the graphic display function, and need accommodate only to a single scheme for controlling the display of information. Second, the presentation system provides a place to merge information from different sub-systems into a single display.

2.1 Information Presentation and Knowledge Representation

As mentioned above, one of the reasons for information presentation is that most of the problems of display construction have been solved and solved again in numerous application dependent forms. The really new problem, indeed the central problem of information presentation, is that of describing the semantic content of a display. Specifically, in what language should we characterize the information to be presented?

Data records with named fields don't convey enough about how what they describe is related to entities known to the presentation system. Do the data pertain to an animal, a vegetable, or a mineral? A concrete object or an abstraction? These distinctions may prove crucial in deciding how the data are to be presented. For example, a program may wish to display the location of a vehicle, while the presentation system understands that maps are excellent for depicting the locations of physical objects. We need a language that facilitates statements about taxonomic categorization, such as that a vehicle is a physical object.

Along with statements about taxonomic relationship, we will also need to state how parts or attributes of a description relate to the parts or attributes of a more general description. For example, how does the changeable location of a vehicle relate to the fixed location of a geographic feature, and how do they both relate to the more general notion of the location of a physical object?

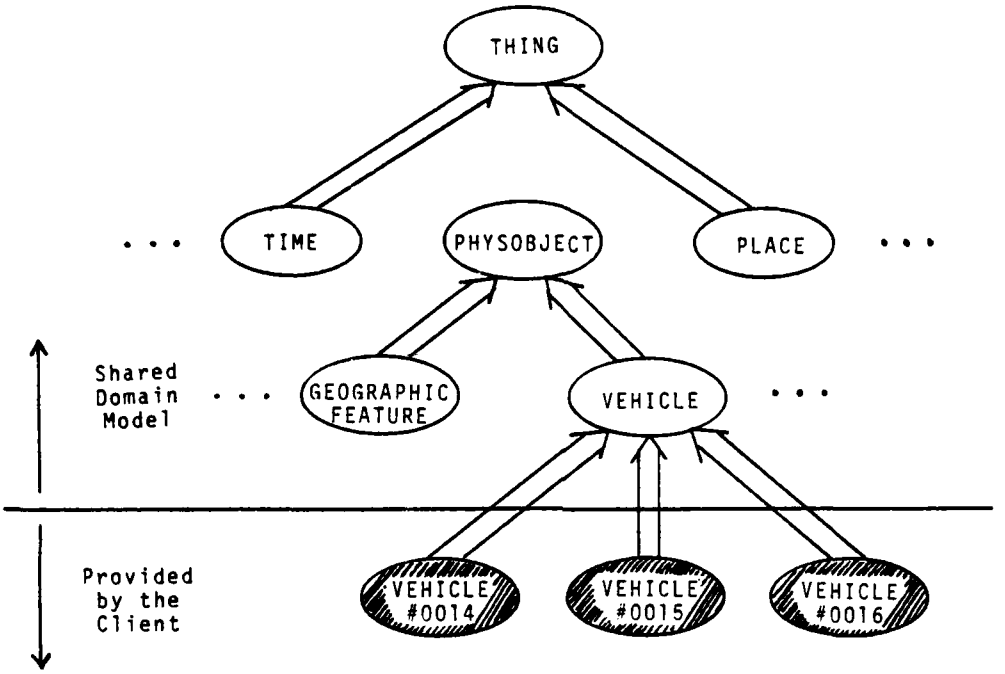
These issues are familiar concerns in the design of what the Artificial Intelligence community terms "knowledge representation languages". In view of this, the Advanced Information Presentation System (AIPS) [Yonke et al., 1980b, Zdybel et al., 1980b, Zdybel et al., 1981a] which this paper describes uses the knowledge representation language KL-ONE [Brachman, 1979, Brachman et al., 1979], also developed at BBN, to describe

the information content of a display. KL-ONE is based on the idea of "Structured Inheritance Nets" [Brachman, 1977], wherein structured descriptions are organized in lattice-like networks, with inheritance of properties among descriptions being carried by structured **Cables**.

KL-ONE descriptions (called **Concepts**) come in two varieties: **Generic Concepts**, which describe categories of objects, and **Individual Concepts**, which describe individual objects. A Cable connecting two Generic Concepts establishes one as a sub-category of the other. A Cable connecting an Individual Concept to a Generic Concept establishes the former as a member of the latter. Thus, a constellation of KL-ONE descriptions connected together with Cables comprises a taxonomic model.

A client system expresses the content of a desired display to AIPS in terms of a shared taxonomic **domain model**. This domain model expresses that knowledge about the world which AIPS must have in order to make decisions about display structure. Figure 1 illustrates a portion of an example domain model. The Generic Concept **THING** is a topological place-holder; it is the distinguished top of all KL-ONE lattices and does not have any internal structure. The Cables (drawn as hollow arrows) leading down to **THING**'s descendants do not imply an exhaustive partition: the model is open-ended in extent as well as in detail. In this example, a set of Individual Concepts (shaded ellipses) could designate a set of vehicles to be included in, perhaps, a map.

FIG. 1. A TAXONOMIC DOMAIN MODEL



Note that if a client system wants to depict some descriptions which are not recognizable in terms of the shared model, it must either expand the model or else re-describe the information in more general, mutually understandable terms. This is the price of AIPS' generality: clarity on the part of the client system in terms of a model shared between the two.

Obviously, it is much easier to connect AIPS to a system which already uses KL-ONE for its internal representations. However, it is certainly possible to translate a data record into a KL-ONE description, and much simpler to do that than to generate a display of the same data.

2.2 KL-ONE Described

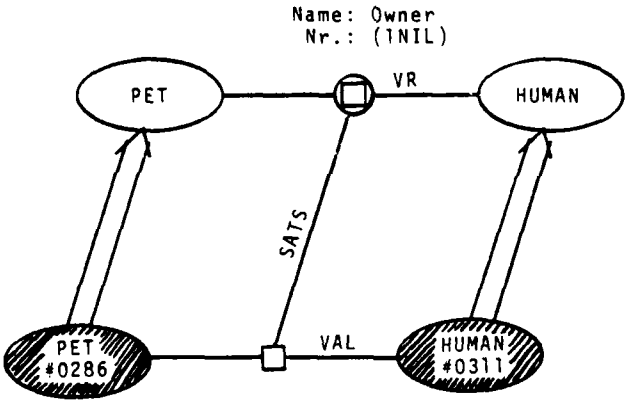
Because KL-ONE plays such an important role in AIPS, we will describe it here briefly before launching into further discussion of AIPS' internal construction. This description is not complete by any means. It is only intended to introduce those aspects of KL-ONE which are of some importance to an overview of AIPS.

One of the things distinguishing KL-ONE from most of the associative network formalisms which preceded it is that a Concept (a network node) has an internal structure of its own. It is, in effect, a small chunk of network in its own right.

The principal element of Concept sub-structure is the **Role**. A Role describes a conceptually identifiable constituent of a Concept. Its structure includes a name (usually used to suggest the function that the constituent is to play in the conceptual complex), a description (called the **ValueRestriction** or **VR**) of the set of legal potential fillers for that Role in any Individual Concept subsumed by the generic description, and a restriction on the number of eventual fillers.

The term "filler" refers to an Individual Concept which satisfies a Role in the structure of some other Individual Concept. Figure 2 shows an example of Role structure at a Generic Concept and how a Role is satisfied at a subsumed Individual Concept. The Role node is drawn as a circle with a square inscribed. The connector to the filler (called an IRole) is drawn as a shaded square.

FIG. 2. A ROLE AND ROLE FILLER



The Cables which interconnect KL-ONE Concepts control the inheritance of parts or attributes among the descriptions: a Cable passes Roles from the more general Concept to the one which sub-categorizes it. There are several ways in which a Role can be inherited (unmodified, modified, or differentiated into several sub-Roles). Because of this, and because the number of

Roles at a Concept is not constrained, there cannot be a single "ISA"-type link (as in "A Dog ISA Mammal") to carry the inheritance. Instead, the Cable has (at least) one internal wire for each Role, specifying that Role's particular relationship to its parent(s) -- hence the names "Cable" and "Structured Inheritance Network".

Since Cables are self-contained, structured entities in and of themselves, KL-ONE Concepts can have multiple ancestors (superCs). Each Cable carries its own information about the superC and its Roles, and therefore multiple Cables will not inadvertently interact. The explicit Role inheritance avoids the problem of "slot" naming confusions common to most network formalisms. In addition to multiple superCs, KL-ONE allows for multiple Role parents and provides for describing the interrelationships between Roles inherited from different parents.

It's important to emphasize that KL-ONE Concepts are descriptions: the same entity in the world may be described by more than one Individual Concept. Existential assertions (e.g. "Joe's pet and the cat on the sofa both exist and are the same thing") are expressed by attachment of Individual Concepts to entities called **Nexuses**, which stand for previously identified (or potentially identifiable) extensional objects. This way the dichotomy between assertion and definition is kept clean, and the intensional world of the lattice is kept homogeneously descriptive.

A critical feature of KL-ONE is that metadescriptions (interpretable comments expressed in KL-ONE) can be attached to any Concept, Role, or Cable. This capability for expressing "knowledge about knowledge" permits the representation of assertions that are awkward or impossible in the base language alone. For example, metadescriptions could be attached to the Cables descending from a Generic Concept to indicate that the sub-categories comprised an exhaustive partition. As explained below, metadescription is used in AIPS to indicate the information contained in a Display.

In addition to metadescriptions, arbitrary LISP objects (KL-ONE is implemented in Interlisp-10 [Teitelman et al., 1978]) can be attached to the network as uninterpreted "tags". Two types of tags are provided for: those that are inheritable at inferior points in the taxonomic lattice and those that are not. The principal importance of this feature to AIPS is that it allows LISP procedures to be attached to KL-ONE descriptions.

2.3 Presentation System Architecture

KL-ONE's role in AIPS is not restricted to representing the content of displays. In fact, a great deal of AIPS' current implementation is expressed in terms of KL-ONE descriptions. AIPS uses KL-ONE to support an object oriented architecture, in a way which facilitates the system's control from the outside.

In a function-oriented programming language, such as LISP, procedural knowledge is associated with a lexical item: a function name. The function definition is uniformly applied each time the function is called, and it applies correctly only to a restricted set of program objects (the definition of the function makes certain assumptions about the nature of the arguments). In an object oriented scheme, procedural knowledge is associated both with a lexical key and some category of objects. Differing procedural knowledge can be associated under the same lexical key at different categories. In effect, the function expression becomes a message passed to an object, and the procedural semantics acted out depend not only on the lexical key of the message, but on the category of the recipient.

Object relative procedure definition is advantageous whenever a procedure must apply to a large or indefinite number of object categories, and the procedural semantics varies from category to category. It is especially useful when a procedure must be easily extensible to apply to additional object categories. Such situations commonly occur in both computer simulation and computer graphics. In a simulation model, many different types of objects must know different methods for updating their state when the simulation clock is incremented. In a graphics system, many different types of objects must know different methods for drawing themselves. In both applications, flexibility derives from being able to extend the set of object categories dealt with by the system.

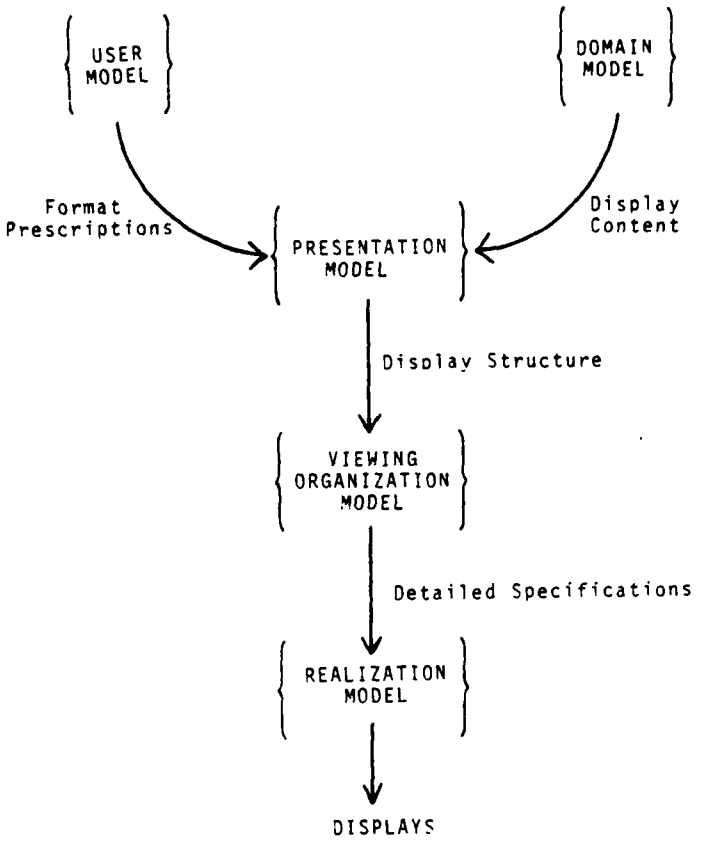
That is why AIPS relies heavily on KL-ONE for its internal structures: KL-ONE facilitates the description of object categories. An object sub-category expressed as a KL-ONE Concept inherits the internal structure (and attached procedures) of the Generic Concept it sub-categorizes. The new description need only express how the sub-category differs from or expands upon the description (and behavior) of the superC. Specific objects of the category are then implemented as Individual Concepts, which automatically inherit the structure and behavior of their containing category.

A further important benefit of AIPS' heavy internal use of KL-ONE is that the presentation system becomes more amenable to outside control. Procedurally expressed knowledge is difficult to modify. To the extent that system behavior is described declaratively, it becomes possible to modify that behavior by changing the associated descriptions. Thus, internal use of KL-ONE by AIPS allows client systems to delegate the display generation function while still retaining considerable control over it.

Figure 3 overviews AIPS' internal organization. Because AIPS is a knowledge-based system, this description concerns the different categories of knowledge important to AIPS, their interactions, and their functions. These collections of knowledge generally are not as modular or isolable as depicted here. The interactions among them are too dense, and the

connections among them too rich for any of them to be fully understood independently. Rather, this factorization is an organizational ideal of the system's designers.

FIG. 3. AIPS' INTERNAL ORGANIZATION



The first category of knowledge, which we have already discussed, is the "domain model." This is a collection of

descriptions of the domain that is to be depicted, in terms of which the desired content of a display is specified to AIPS, and in terms of which AIPS understands how display structure depends upon semantic content. The declarative nature of the domain model helps with the problem of connecting AIPS simultaneously to several client systems, because it provides a medium in which redundant or conflicting information can be separately represented and explicitly reconciled.

The user model contains information about AIPS' human user. In AIPS' current incarnation this information is expressed primarily in terms of format structure and how the user's preferences differ from AIPS' default assumptions. Such information can range widely in function within AIPS. As a result, AIPS' current user model tends to be distributed over the system in the form of qualifications and amendments to what the system otherwise knows about preparing displays. In a sophisticated information presentation system, the user model might include explicit descriptions of the display end-consumer and his perceptual and decision tasks. This would allow the system to modify its decisions about display structure and content according to the user's momentary needs.

The third important category of AIPS knowledge, the presentation model, describes display formats such as "map", "graph", "table", "legend" and "menu." These descriptions, via reference to the domain model, include information about how

format structure and suitability depend on the semantic content to be presented. The function of this knowledge is to convert content specifications into a skeletal structure which describes the component parts and pieces of the eventual display and their functional relationships to each other.

The viewing organization model deals with activities such as scaling, clipping, and transformation. The important distinction between it and the presentation model is that here the display and its constituents are treated as geometric and topological entities, without reference to their semantic content. The descriptions in this model include concepts such as display windows, clipping regions, coordinate systems, and mappings. These descriptions and the procedures attached to them fill in the locational details of the relational structure produced by the presentation model, and complete the description of the display.

Although display layout is handled procedurally in the current version of AIPS, the viewing organization model should eventually include declaratively expressed constraints on the positioning of display constituents. Some issues that could be treated in this way include positioning labels in maps, expressing the layout of a table, or constraining two display windows to be adjacent. A declarative treatment of these issues would make the human factors of display explicit and accessible.

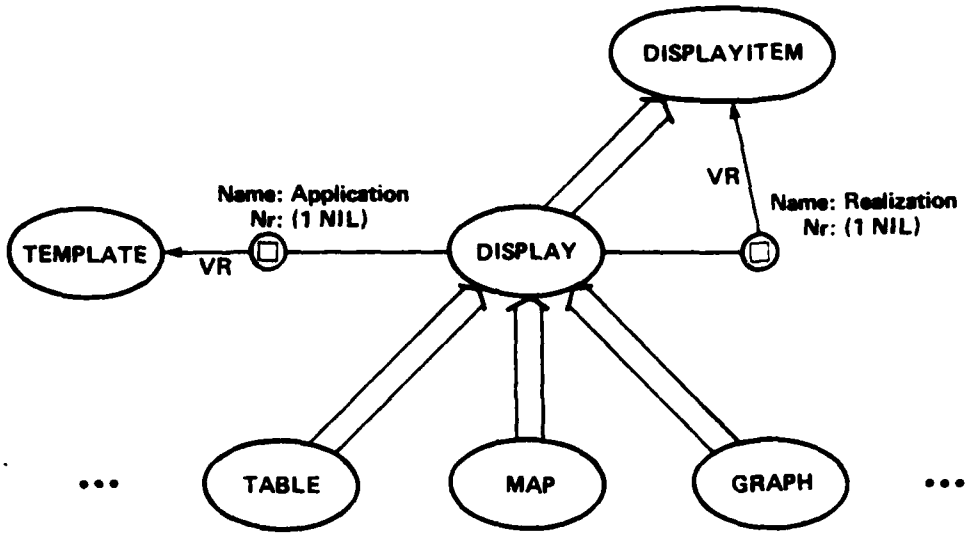
All that remains is to draw the display, and that is the function of the realization model. Most of the descriptions in this model concern such things as characters, points, lines, arcs and simple geometric shapes. The procedural knowledge attached to the descriptions tells how to draw and erase these entities in device-dependent terms. Display devices are also described declaratively in this model, so that device characteristics can influence the display planning and layout processes. For example, a description of the size and shape of the display screen can be used to limit the size of a map.

2.4 Display Description Structure

For the sake of clarity, we will adhere in this discussion to a certain typographical convention. The names of KL-ONE Generic Concepts are printed entirely in bold capitals. The names of Individual Concepts are simply capitalized (where the name is a compound word, sub-words will also be capitalized). The names of Roles are capitalized and underscored. Thus, **DISPLAY** refers to the description of a category, Display refers to an individual of that category, and display refers to something seen on the screen of a graphics terminal.

As shown in Figure 4, various display formats are represented as sub-categories of **DISPLAY**, which is the top-level

FIG. 4. DISPLAY AND ITS SUB-CATEGORIES

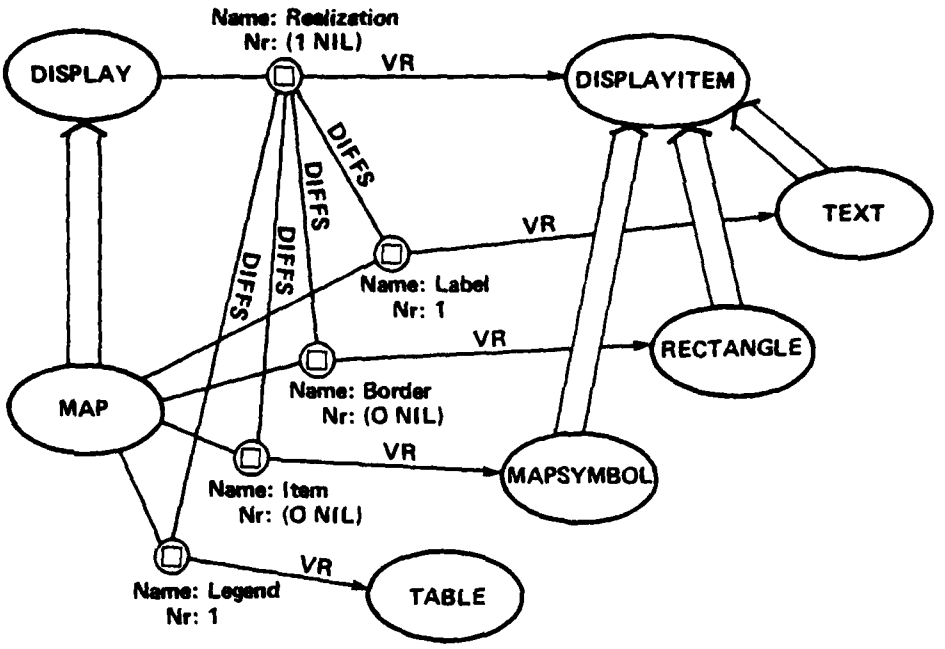


Concept of the presentation model. There are two important themes in each such description: a characterization of the information involved in the display, and a characterization of its visible components. These are represented respectively by the Application and Realization Roles of DISPLAY.

The Realization Role is differentiated (split into multiple sub-Roles) and modified at the various descendants of **DISPLAY**. For example, **MAP**'s visible components: Border, Label, Legend, Item, etc. are all represented as differentiating sub-Roles of **DISPLAY**'s Realization Role (See Figure 5).

Notice that the ValueRestriction of Realization is the concept **DISPLAYITEM**, which is a superC of DISPLAY. DISPLAYITEM

FIG. 5. THE INTERNAL STRUCTURE OF A FORMAT'S REALIZATION



is an important upper node of the viewing organization model. It provides a characterization of a piece of a display in purely graphic terms. Attribute Roles of **DISPLAYITEM** relate such things as the location, orientation, scale, width and height of a display element. Because **DISPLAYITEM** is a superC of **DISPLAY**, the eventual fillers of (sub-Roles of) the Realization role in any Display may be either simply treated syntactically (as

DisplayItems) or may in fact be Displays in their own right, carrying an explicit treatment of the information they depict. Also, of course, any Display inherits all of the syntactic attributes of **DISPLAYITEM**, and thus can be described in terms of attributes such as location, width, height, etc.

The Application Role of **DISPLAY** indicates the information being expressed in a given Display. This Role is not currently modified or expanded upon at any of **DISPLAY**'s descendants; it is simply filled for individual Displays. The filler(s) of the Application Role do not necessarily describe all of the information involved in a Display. Rather, this Role is a kind of binding mechanism that characterizes a Display's application to specific information, as opposed to the inherent use of information that might be specified in its generic definition (e.g. a sub-category of **MAP** which always labels items with their names, regardless of whether or not that attribute is mentioned at the Application.)

Of course, a sub-category of **DISPLAY** may add descriptive Roles that are sub-Roles of neither Realization nor Application. For example, **MAP** may have a Role Injection which characterizes the scaling transformation for a map.

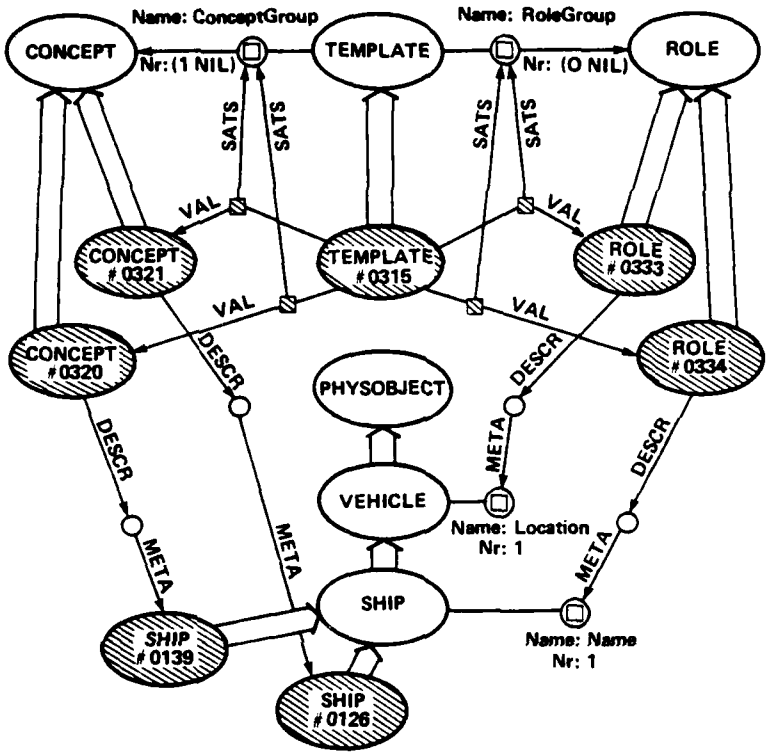
2.5 The Characterization of Information

At **DISPLAY**, the Application Role is value restricted to one or more individuals of category **TEMPLATE**. A Template is simply a means of indicating the cross product of some set of descriptions with some subset of their Roles. Thus, **TEMPLATE** has two roles: ConceptGroup and RoleGroup. The fillers of ConceptGroup indicate the domain model objects of concern in a Display; the fillers of RoleGroup indicate which attributes of those objects are involved. If the information content of a display does not factor into a single such cross product, several Templates may be used to capture the disjuncts.

A Template gets its necessary descriptive "grip" on Concepts and Roles in the domain model through use of KL-ONE's meta-description feature. The Roles to be included in a Template are linked by **description** wires to Nexuses which are then meta-hooked to Individual Concepts of the meta-Concept **ROLE**. These individuals become the fillers of the RoleGroup of the Template. Meta-hooks are used in a similar way to indicate the Concepts involved in a Template via the meta-Concept **CONCEPT**. See Figure 6 for an example of a Template used to meta-indicate the names and locations of two ships.

Templates are also useful as meta-descriptions of display formats. For example, a Template which refers to a Generic Concept of the domain model can be used to indicate what

FIG. 6. HOW A TEMPLATE META-INDICATES INFORMATION



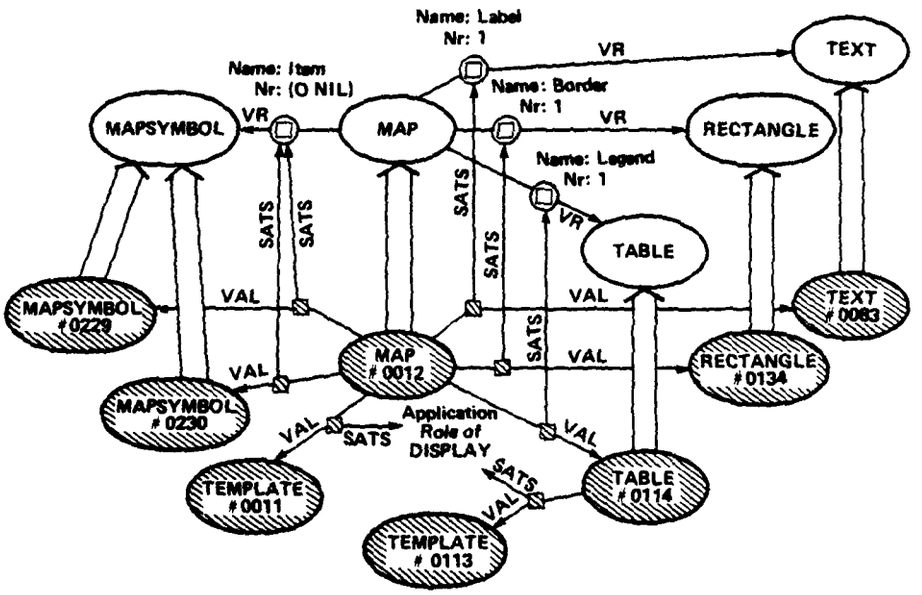
information must be known about objects of that category in order for them to be depicted in a particular format.

2.6 Procedural Knowledge of the Presentation System

The process of creating a display begins by creating a "blank" individual (i.e. none of its Roles are yet filled) of

some sub-category of **DISPLAY** and filling its Application with the Template or Templates specified by the client. If the client does not specify the desired format, AIPS compares the Template with meta-descriptions on the formats it knows about and selects a suitable category. From this point, display construction proceeds in three phases: derivation, location, and drawing. Each phase is supported by its own set of procedural attachments to the knowledge base.

FIG. 7. THE DESCRIPTION OF AN INDIVIDUAL DISPLAY



During the derivation phase, the description of the Display is expanded at least to the extent that all sub-Roles of Realization are filled with some DisplayItem; each of these descriptions is then recursively expanded in the same manner. For example, for the case shown in Figure 7, the Legend Role of the Map being constructed is filled with an individual of **TABLE** whose Application Role filler indicates the set of symbols used in the map. The derivation process then recurses on this Table to fill out its internal structure.

The derivation phase procedures are attached as tags to the Roles for which they produce fillers. These tags specify what information must already be known in order to run a procedure, so that a demand to fill a Role may first result in the filling of other Roles on which its derivation depends. Also, the body of an attached derivation procedure can dynamically call for the derivation of some other Role and suspend processing until the information is provided. Derivation procedures are inherited and more than one derivation procedure may be attached. At Role filler derivation time, all of the available procedures are tried in order from the more specific to the more general until one succeeds.

The second or location phase proceeds by means of messages passed among the constituent objects of the Display, which were identified and constructed in the preceding derivation phase. DisplayItems receive **ToLocate** messages which tell them where they

are located relative to the coordinate system of the Display's viewing surface (which entities by now have also both been completely described). If the recipient DisplayItem contains separately described components, the attached ToLocate procedure computes the locations of these constituents and recurses the location process by dispatching further ToLocate messages.

The final or drawing phase is handled in a similar manner. DisplayItems receiving ToDraw messages execute ToDraw procedures which ultimately call the drawing routines of a graphics package. DisplayItems with separately described components send further ToDraw messages.

2.7 Conclusions

The version of AIPS described here runs on a DecSystem-20 under the Interlisp-10 interpreter, using a bitmap graphics terminal of BBN's design. The BMG graphics language [Greenfeld et al, 1980] used in this work was developed and implemented as a part of the AIPS effort, which also made substantial contributions to the current implementation of KL-ONE. Our research is supported by the Defense Advanced Research Projects Agency (DARPA) Information Processing Technology Office.

AIPS was conceived as a display management tool suitable for work environments supported by fast personal machines with very

large virtual memories, such as the MIT CADR machine [Greenblatt, 1978], Xerox PARC's Dorado [Lampson and Pier, 1980] or BBN's Jericho [Greenfeld, 1981]. The current AIPS is a carefully delimited prototype which barely fits into Interlisp-10's available storage. We are presently much occupied with moving AIPS (and the Interlisp environment which supports it) onto the Jericho symbolic processor. When that has been accomplished, we will be in a position to further elaborate the structure of AIPS' display descriptions.

Many of the benefits of information presentation can be had with less powerful and more widely available tools than KL-ONE and LISP, albeit at the cost of some generality. The broadest importance of this work is simply that it demonstrates one method of raising the level of interaction between a program and its graphic display function. We are not alone in this pursuit. What distinguishes AIPS is its direct assault on the inherent knowledge representation issues.

REFERENCES

Brachman, R.J. A Structural Paradigm for Representing Knowledge. PhD thesis, Harvard University, May, 1977. Also, BBN Report No.3605, Bolt Beranek and Newman Inc., May 1978.

Brachman, R.J. On the epistemological status of semantic networks. Pages 3-50. In N.V.Findler (Ed.), Associative Networks --the Representation and Use of Knowledge in Computers. New York: Academic Press, 1979.

Brachman, R.J., Bobrow, R.J., Cohen, P.R., Klovstad, J.W., Webber, B.L., Woods, W.A. Research in Natural Language Understanding - Annual Report: 1 Sept 78 - 31 Aug 79. BBN Report No. 4274, Bolt Beranek and Newman Inc., August 1979.

Greenblatt, R., Knight, T., Holloway, J., Moon, D. The LISP Machine. Massachusetts Institute of Technology Artificial Intelligence Laboratory 1978.

Greenfeld, N.R. Jericho: A Professional's Personal Computer System. Proceedings of the 8th Annual Symposium on Computer Architecture. IEEE Computer Society and the Association for Computing Machinery, Minneapolis, Mn., May 1981.

Greenfeld, N.R., Zdybel, F., Ciccarelli, E., and Yonke, M.D. BMG Reference Manual. BBN Report No. 4368, Bolt Beranek and Newman Inc., April 1980.

Lampson, B.W.,and Pier, K.A. A Processor for a High-Performance Personal Computer. Proceedings of the 7th Annual Symposium on Computer Architecture. The IEEE Computer Society and the Association for Computing Machinery, Le Baule, France, May 1980.

Teitelman, W. A Display Oriented Programmer's Assistant, pages 905-915. Proceedings of the Fifth International Joint Conference on Artificial Intelligence. IJCAI, Cambridge, MA, 1977.

Teitelman, W., Goodwin, J.W., Hartley, A.K., Lewis, D.C., Vittal, J.J., Yonke, M.D., Bobrow, D.G., Kaplan, R.M., Masinter, L.M., Sheil, B.A. INTERLISP Reference Manual. Revised October 1978 edition, Xerox Palo Alto Research Center, Palo Alto, CA, 1978.

Yonke, M.D., Greenfeld N.R. An Information Presentation System For Decision Makers. Data Base, Fall 1980, 12(1/2), pages

26-32. Selected papers on Decision Support Systems from the 13th Hawaii International Conference on System Sciences, Ralph H.Sprague, Jr., Editor.

Yonke, M.D., and Greenfeld, N.R. AIPS: An Information Presentation System for Decision Makers, pages 48-56. Presented at the Thirteenth Hawaii International Conference on System Sciences, University of Hawaii, January 1980. Also reprinted in Data Base, V.12, 1980. A revised version of this paper is also available as BBN Report No. 4228, Bolt Beranek and Newman Inc., December 1979.

Zdybel, F., First Annual National Conference on Artificial Intelligence. Panel discussion on Computing and Programming Environments, with Bob Balzer, Peter Deutsch, Scott Fahlman, Edward Feigenbaum, and Richard Greenblatt, Stanford University, August 19-21, 1980.

Zdybel, F., Yonke, M.D., and Greenfeld, N.R. Application of Real-Time Symbolic Processing to Command and Control, Final Technical Report. BBN Report No. 3849, Bolt Beranek and Newman Inc., February 1980.

Zdybel, F., Greenfeld, N.R., and Yonke, M.D. Application of Symbolic Processing to Command and Control: An Advanced Information Presentation System, Annual Technical Report. BBN Report No. 4371, Bolt Beranek and Newman Inc., April 1980.

Zdybel, F., Greenfeld, N., Yonke, M., Gibbons, J. An Information Presentation System, pages 978-984. Proceedings of the Seventh International Joint Conference on Artificial Intelligence. IJCAI, Vancouver, B.C., August 1981.

Zdybel, F., Greenfeld, N., Yonke, M., Gibbons, J. An Information Presentation System, pages-. Proceedings of the Computer Graphics 81 International Conference and Exhibition. Online Conferences, London, U.K., October 1981.